

Next.js + Prisma

On your command line:

- `npx create-next-app@latest <name_of_your_application>`
 - o Select all the defaults
- `npm run dev`
 - o Go to `localhost:3000` in your browser to ensure the application is running
- `npm install prisma --save-dev`
- `npm install @prisma/client`
- `npx prisma init`

In the `.env` file

- Update the `DATABASE_URL` with the url of your prisma instance, which can be found from last weeks project or from the prisma.io website using the database you set up.

On your command line:

- `npx prisma migrate reset`

In the schema.prisma file

- Update the schema to be relevant to your project. For example, the grocery list project schema looks like this

```
prisma > schema.prisma > ...
Generate
1 generator client {
2   provider = "prisma-client-js"
3   output   = "../generated/prisma"
4 }
5
6 datasource db {
7   provider = "postgresql"
8   url      = env("DATABASE_URL")
9 }
10
11 model GroceryList {
12   id      Int      @id @default(autoincrement())
13   name    String
14   items   GroceryListItem[]
15 }
16
17 model GroceryListItem {
18   id          Int      @id @default(autoincrement())
19   name        String
20   purchased   Boolean  @default(false)
21   groceryListId Int
22   groceryList GroceryList @relation(fields: [groceryListId], references: [id])
23 }
24 | %%L to chat, %%K to generate
```

Once your changes are made, on the command line:

- npx prisma migrate dev --name init

In the app folder, create an api folder. In that folder, create a route.ts file and add this code

```
app > api > TS route.ts > ...
1   export async function GET() {
2     return new Response('Hello from Next.js!', {
3       status: 200,
4       headers: { 'Content-Type': 'application/json' }
5     });
6   }
7   ⌘L to chat, ⌘K to generate
```

In your browser, go to localhost:3000/api. You should see “Hello from Next.js”.

That’s all! You now have a working database and backend. Next week we will connect the two!